

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/261557716>

Information Theory: A Tutorial Introduction

Book · February 2015

DOI: 10.13140/2.1.1633.8240

CITATIONS

38

READS

10,379

1 author:



James V Stone

The University of Sheffield

82 PUBLICATIONS 2,140 CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:



Neural Information Theory (BOOK) [View project](#)

Information Theory

A Tutorial Introduction

James V Stone

Information Theory:
A Tutorial Introduction
Author: James V Stone
Published by Sebtel Press

All rights reserved. No part of this book may be reproduced or transmitted in any form without written permission from the author.

Copyright ©2014 by James V Stone
First Edition, 2014.
Typeset in L^AT_EX 2_ε.
File: infotheory_book_v19.tex, July 17, 2014.
ISBN 978-0956372857

For Nikki

Suppose that we were asked to arrange the following in two categories - *distance*, *mass*, *electric force*, *entropy*, *beauty*, *melody*. I think there are the strongest grounds for placing entropy alongside beauty and melody . . .

Eddington A, *The Nature of the Physical World*, 1928.

Contents

Preface

1. What Is Information?	1
1.1. Introduction	1
1.2. Information, Eyes and Evolution	2
1.3. Finding a Route, Bit By Bit	3
1.4. A Million Answers to Twenty Questions	8
1.5. Information, Bits and Binary Digits	10
1.6. Example 1: Telegraphy	11
1.7. Example 2: Binary Images	13
1.8. Example 3: Grey-Level Images	15
1.9. Summary	20
2. Entropy of Discrete Variables	21
2.1. Introduction	21
2.2. Ground Rules and Terminology	21
2.3. Shannon's Desiderata	29
2.4. Information, Surprise and Entropy	30
2.5. Calculating Entropy	37
2.6. Properties of Entropy	39
2.7. Independent and Identically Distributed Values	41
2.8. Bits, Shannons, and Bans	41
2.9. Summary	42
3. The Source Coding Theorem	43
3.1. Introduction	43
3.2. Capacity of a Discrete Noiseless Channel	44
3.3. Shannon's Source Coding Theorem	47
3.4. Calculating Information Rates	48
3.5. Data Compression	52
3.6. The Entropy of English Letters	59
3.7. Why the Source Coding Theorem is True	68
3.8. Kolmogorov Complexity	73
3.9. Summary	73

4. The Noisy Channel Coding Theorem	75
4.1. Introduction	75
4.2. Joint Distributions	76
4.3. Mutual Information	85
4.4. Conditional Entropy	88
4.5. Noise and Cross-Talk	91
4.6. Noisy Pictures and Coding Efficiency	94
4.7. Adding Redundancy: Error Correcting Codes	97
4.8. Capacity of a Noisy Channel	100
4.9. Shannon's Noisy Channel Coding Theorem	101
4.10. Why the Noisy Coding Theorem is True	105
4.11. Summary	106
 5. Entropy of Continuous Variables	 107
5.1. Introduction	107
5.2. The Trouble With Entropy	108
5.3. Differential Entropy	111
5.4. Under-Estimating Entropy	113
5.5. Properties of Differential Entropy	114
5.6. Maximum Entropy Distributions	116
5.7. Making Sense of Differential Entropy	123
5.8. What is Half a Bit of Information?	124
5.9. Summary	128
 6. Mutual Information: Continuous	 129
6.1. Introduction	129
6.2. Joint Distributions	131
6.3. Conditional Distributions and Entropy	135
6.4. Mutual Information and Conditional Entropy	139
6.5. Mutual Information is Invariant	143
6.6. Kullback-Leibler Divergence	144
6.7. Summary	146
 7. Channel Capacity: Continuous	 147
7.1. Introduction	147
7.2. Channel Capacity	147
7.3. The Gaussian Channel	148
7.4. Error Rates of Continuous Noisy Channels	154
7.5. Using a Gaussian Channel	156
7.6. Mutual Information and Correlation	157
7.7. The Fixed Range Channel	159
7.8. Calculating Mutual Information	166
7.9. Summary	167

8. Thermodynamic Entropy and Information	169
8.1. Introduction	169
8.2. Physics, Entropy and Disorder	170
8.3. Information and Thermodynamic Entropy	172
8.4. Ensembles, Macrostates and Microstates	174
8.5. Pricing Information: The Landauer Limit	176
8.6. The Second Law of Thermodynamics	178
8.7. Maxwell's Demon	179
8.8. Summary	182
9. Could It Have Been Any Different?	183
9.1. Introduction	183
9.2. Satellite TVs, MP3 and All That	183
9.3. Does Sex Accelerate Evolution?	186
9.4. The Human Genome: How Much Information?	191
9.5. Are Brains Good at Processing Information?	192
9.6. A Very Short History of Information Theory	200
9.7. Summary	200
Further Reading	201
A. Glossary	203
B. Mathematical Symbols	211
C. Logarithms	215
D. Probability Density Functions	217
E. Averages From Distributions	219
F. The Rules of Probability	221
G. The Gaussian Distribution	225
H. Key Equations	227
References	234
Index	235

Preface

This book is intended to provide a coherent and succinct account of information theory. Inevitably, understanding information theory requires a degree of mathematical sophistication. However, in order to develop an intuitive understanding of key ideas, new topics are presented in an informal tutorial style, accompanied by diagrams, before being described more formally. The equations which underpin the mathematical foundations of information theory are introduced on a ‘need to know basis’, and are accompanied with sufficient explanatory text to ensure their meaning is made clear.

In mathematics, rigour follows insight, and not *vice versa*. Kepler, Newton, Fourier and Einstein developed their theories from deep intuitive insights into the structure of the physical world, which requires, but is not motivated by, the raw logic of pure mathematics. In a similar vein, it is hoped that this book provides insights into how information theory works, and why it works in that way. This is entirely consistent with Shannon’s own approach. In his famously brief book, Shannon prefaced his account of information for continuous variables with these words,

We will not attempt in the continuous case to obtain our results with the greatest generality, or with the extreme rigor of pure mathematics, since this would involve a great deal of abstract measure theory and would obscure the main thread of the analysis. . . . The occasional liberties taken with limiting processes in the present analysis can be justified in all cases of practical interest.

In a similar vein, one of the fathers of modern probability theory protested that,

... the more one concentrates on the appearance of mathematical rigor, the less attention one pays to the validity of the premises in the real world, and the more likely one is to reach final conclusions that are absurdly wrong in the real world.

ET Jaynes and GL Bretthorst,

Probability Theory: The Logic of Science²¹, 2003.

While this is no excuse for incorrect or sloppy mathematics, it is a clear recommendation that we should not mistake a particular species of pedantry for mathematical rigour. The spirit of this liberating and somewhat cavalier approach is purposely adopted in this book, which is intended to provide insights, rather than incantations, regarding how information theory is relevant to problems of practical interest.

MatLab and Python Computer Code

It often aids understanding to be able to examine well documented computer code, which provides an example of a particular calculation or method. Accordingly, MatLab and Python code which implements key information theoretic methods, and which also produces several figures in this book, can be found online.

MatLab code can be downloaded from here:

<http://jim-stone.staff.shef.ac.uk/BookInfoTheory/InfoTheoryMatlab.html>

Python code can be downloaded from here:

<http://jim-stone.staff.shef.ac.uk/BookInfoTheory/InfoTheoryPython.html>

Powerpoint Slides of Figures

Most of the figures used in this book are available for teaching purposes as a pdf file and as powerpoint slides. These can be downloaded from <http://jim-stone.staff.shef.ac.uk/BookInfoTheory/InfoTheoryFigures.html>

Corrections

Please email any corrections to j.v.stone@sheffield.ac.uk.

A list of corrections can be found at

<http://jim-stone.staff.shef.ac.uk/BookInfoTheory/Corrections.html>

Acknowledgments

Thanks to John de Pledge, John Porrill, Royston Sellman, and Steve Snow for interesting discussions on the interpretation of information theory, and to John de Pledge for writing much of the Python code, which can be downloaded from the book's web site (see above).

For reading draft chapters, I am very grateful to David Buckley, Stephen Eglen, Nikki Hunkin, Guy Mikawa, Xiang Mou, Oscar Perez, and Stuart Wilson.

Jim Stone,
Sheffield, England,
July 2014.

Chapter 1

What Is Information?

Most of the fundamental ideas of science are essentially simple, and may, as a rule, be expressed in a language comprehensible to everyone.

Einstein A and Infeld L, *The Evolution of Physics*, 1938.

1.1. Introduction

The universe is conventionally described in terms of physical quantities such as mass and velocity, but a quantity at least as important as these is *information*. Whether we consider computers²⁵, evolution^{2;17}, physics¹⁴, artificial intelligence⁹, or brains^{15;38}, we are driven to the inescapable conclusion that their behaviour is determined by the way they process information.



Figure 1.1.: Claude Shannon (1916-2001).

1 What Is Information?

In 1948, Claude Shannon published a paper called, “*A Mathematical Theory of Communication*”⁴². The publication of this paper heralded a transformation in our conception of information. In the centuries before Shannon’s paper was published, information had been viewed as a kind of poorly defined miasmic fluid. After Shannon’s paper, it became apparent that information is a well defined, and above all, *measurable* quantity.

Shannon’s paper describes a subtle theory, which tells us something fundamental about the way the universe works. However, unlike other great theories, such as the Darwin-Wallace theory of evolution, information theory is not simple, and it is full of caveats. But we can disregard many of these caveats provided we keep a firm eye on the physical interpretation of information theory’s defining equations. This will be our guiding principle in exploring the theory of information.

1.2. Information, Eyes and Evolution

Shannon’s theory of information provides a mathematical definition of information, and describes precisely how much information can be communicated between different elements of any system. This may not sound like much, but Shannon’s theory underpins our understanding of how signals and noise are related, and why there are definite limits to the rate at which information can be communicated within *any* system, man-made or biological. It represents one of the few examples of a single theory being responsible for creating an entirely new field of research. In this regard, Shannon’s theory should rank alongside those of Darwin-Wallace, Newton, and Einstein.

When a question is typed into a computer search engine, the results provide useful information, but this is buried in a sea of mostly useless data. So in this internet age, it is easy for us to appreciate the difference between information and mere data, and we have learned to treat the information as a useful ‘signal’ and the rest as distracting ‘noise’. This experience is now so commonplace that technical phrases like ‘signal to noise ratio’ are becoming part of everyday language. Even though most people are unaware of the precise meaning of this phrase, they have an intuitive grasp of the idea that data consist of a combination of (useful) signal and (useless) noise.

The ability to separate signal from noise, to extract information from data, is crucial for modern telecommunications. For example, it allows a television picture to be compressed down to its bare information bones, transmitted to a satellite, and then to a TV, before being uncompressed to reveal the original picture on a TV screen.

This type of scenario is also ubiquitous in the natural world. The ability of eyes and ears to extract useful signals from noisy sensory data, and to package those signals efficiently, is the key to survival⁴⁵. Indeed, the *efficient coding hypothesis*^{5;8;38;49} suggests that the evolution of sense organs, and of the brains that process data from those organs, is dominated by the need to expend as little energy as possible for each bit of information acquired from the environment. More generally, a particular branch of brain sciences, *computational neuroscience*, relies on information theory to provide a benchmark against which to measure the performance of neurons objectively.

On a grander biological scale, the ability to separate signal from noise is fundamental to the Darwin-Wallace theory of evolution by natural selection¹¹. Evolution works by selecting individuals best suited to a particular environment, so that over many generations, information about the environment gradually accumulates within the gene pool. Thus, natural selection is essentially a means by which information about the environment is incorporated into DNA. And it seems likely that the rate at which information is incorporated into DNA is accelerated by an age-old biological mystery, sex. These, and other ‘applications’ of information theory are described in chapter 9.

1.3. Finding a Route, Bit By Bit

Information is measured in *bits*, and one bit of information allows you to choose between two equally probable alternatives. The word bit is derived from *binary digit* (i.e. a zero or a one). However, as we shall see, bits and binary digits are fundamentally different types of entities.

Imagine you are standing at a fork in the road at point A in Figure 1.2, and that you want to get to the point marked D. Note that this figure represents a bird’s-eye view, which you do not have; all you have is a fork in front of you, and a decision to make. If you have no prior

1 What Is Information?

information about which road to choose then the fork at A represents two equally probable alternatives. If I tell you to go left then you have received one bit of information. If we represent my instruction with a *binary digit* (i.e. 0=‘left’, and 1=‘right’) then this binary digit provides you with one bit of information, which tells you which road to choose.

The decision taken at A effectively excludes half of the eight possible destinations that you could arrive at, shown in Figure 1.2. Similarly, the decision taken at each successive fork in the road effectively halves the number of remaining possible destinations that you could arrive at.

Now, imagine that you stroll on down the road, but then you come to another fork, point B in Figure 1.2. Again, because you have no idea which road to choose, another binary digit (1=right) provides one more bit of information, and allows you to choose the correct road, which leads to the point marked C.

Note that C is one out of 4 possible interim destinations that you could have reached after making two decisions. The two binary digits that allow you to make the correct decisions provided you with two

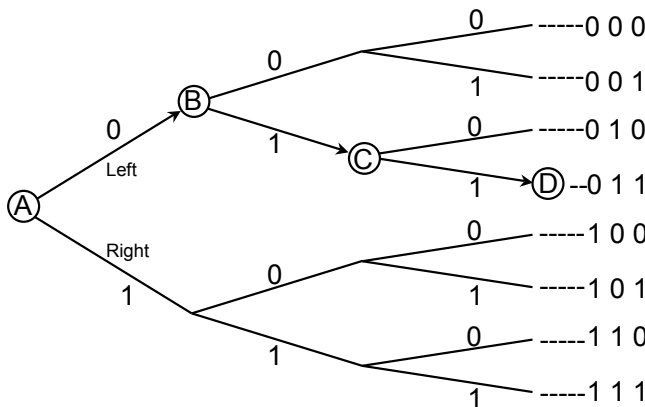


Figure 1.2.: How many roads must a man walk down? For a traveller who does not know the way, each fork in the road requires one bit of information to make a correct decision. The binary (zero/one) numbers on the right hand side summarise the instructions needed to arrive at each destination if a left turn is indicated by a 0 and a right turn by a 1.

bits, and allow you to choose from amongst four (equally probable) possible alternatives, which also happens to be equal to $2 \times 2 = 2^2$.

A third binary digit (1=right) provides you with one more bit of information, which allows you to choose the correct road again, leading to the point marked D.

There are now eight roads you could have chosen from when you started at A, so three binary digits (which provide you with 3 bits) allow you to choose from amongst eight (equally probable) alternatives, which also happens to equal $2 \times 2 \times 2 = 2^3 = 8$.

A Journey of 8 Alternatives

Let's summarise your journey in terms of the number of equally probable alternatives:

If you have 1 bit of information then you can choose between 2 equally probable alternatives (i.e. $2^1 = 2$).

If you have 2 bits of information then you can choose between 4 equally probable alternatives (i.e. $2^2 = 4$).

Finally, if you have 3 bits of information then you can choose between 8 equally probable alternatives (i.e. $2^3 = 8$).

We can re-state this in more general terms if we use n to represent the number of forks, and m to represent the number of final destinations. If you have come to n forks then you have effectively chosen from amongst

$$m = 2^n, \tag{1.1}$$

final destinations. As the decision at each fork requires one bit of information, n forks require n bits of information, which allow you to choose from 2^n equally probable alternatives.

There is a saying that, "a journey of a thousand miles begins with a single step". In fact, a journey of a thousand miles begins with a single decision: the direction in which to take the first step.

Key point. One bit is the amount of information required to choose between two *equally probable* alternatives.

Binary Numbers

We could label each of the eight possible destinations with a decimal number between 0 and 7, or with the equivalent *binary number*, as in Figure 1.2. These decimal numbers, and their equivalent binary representations, are shown in Table 1.1. Counting in binary is analogous to counting in decimal. Just as each decimal digit in a decimal number specifies how many 1s, 10s, 100s (etc) there are, so, each binary digit in a binary number specifies how many 1s, 2s, 4s (etc) there are. For example, the value of the decimal number 101 equals the number of 100s (i.e. 10^2), plus the number of 10s (i.e. 10^1), plus the number of 1s (i.e. 10^0)

$$(1 \times 100) + (0 \times 10) + (1 \times 1) = 101. \quad (1.2)$$

Similarly, the value of the binary number 101 equals the number of 4s (i.e. 2^2), plus the number of 2s (i.e. 2^1), plus the number of 1s (i.e. 2^0)

$$(1 \times 4) + (0 \times 2) + (1 \times 1) = 5. \quad (1.3)$$

The binary representation of numbers has many advantages. For instance, the binary number (e.g. 011) that labels each destination, explicitly represents the set of left/right instructions required to reach that destination. This applies to any problem that consists of making a number of two-way (i.e. binary) decisions.

Logarithms

The complexity of any journey can be represented either as the number of possible final destinations, or as the number of forks in the road. We know that, as the number of forks increases, so the number of possible

Decimal	Binary	Decimal	Binary
0	000	4	100
1	001	5	101
2	010	6	110
3	011	7	111

Table 1.1.: Decimal numbers and their equivalent binary representations.

destinations also increases. As we have already seen, if there are 3 forks then there are $8 = 2^3$ possible destinations.

Viewed from another perspective, if there are $m = 8$ possible destinations then how many forks n does this imply? In other words, given 8 destinations, what power of 2 is required in order to get 8? In this case, we know the answer is $n = 3$, which is called the *logarithm* of 8. Thus, $3 = \log_2 8$ is the number of forks implied by 8 destinations.

More generally, the logarithm of m is the power to which 2 must be raised in order to obtain m ; that is, $m = 2^n$. Equivalently, given a number m , which we wish to express as a logarithm,

$$n = \log_2 m. \quad (1.4)$$

The subscript 2 indicates that we are using logs to the base 2 here (all logarithms in this book have base 2, unless stated otherwise). See Appendix C for a tutorial on logarithms.

A Journey of $\log_2(8)$ Decisions

Now that we know about logarithms, we can summarise your journey from a different perspective, in terms of bits:

If you have to choose between 2 equally probable alternatives (i.e. 2^1) then you need $1 (= \log_2 2^1 = \log_2 2)$ bit of information.

If you have to choose between 4 equally probable alternatives (i.e. 2^2) then you need $2 (= \log_2 2^2 = \log_2 4)$ bits of information.

If you have to choose between 8 equally probable alternatives (i.e. 2^3) then you need $3 (= \log_2 2^3 = \log_2 8)$ bits of information.

More generally, if you have to choose between m equally probable alternatives then you need $n = \log_2 m$ bits of information.

Key point. If you have n bits of information then you can choose from $m = 2^n$ equally probable alternatives. Equivalently, if you have to choose between m equally probable alternatives then you need $n = \log_2 m$ bits of information

1.4. A Million Answers to Twenty Questions

Navigating a series of forks in the road is, in some respects, similar to the game of ‘20 questions’. In this game, your opponent thinks of a word (usually a noun), and you (the astute questioner) are allowed to ask 20 questions in order to discover the identity of this word. Crucially, each question must have a yes/no (i.e. binary) answer, and therefore provides you with a maximum of one bit of information.

By analogy with the previous example, where each decision at a road fork halved the number of remaining destinations, each question should *halve* the number of remaining possible words. In doing so, each answer provides exactly 1 bit of information to you. An example of a poorly chosen question is one to which you already know the answer. For example, if your question is, “Is the word in the dictionary?”, then the answer is almost certainly, “Yes!”, an answer which is predictable, and which therefore provides you with no information.

Conversely, a well chosen question is one to which you have no idea of whether the answer will be yes or no, and in this case, the answer provides exactly one bit of information. This should be more apparent from the cut-down version of 20 questions shown in Figure 1.3.

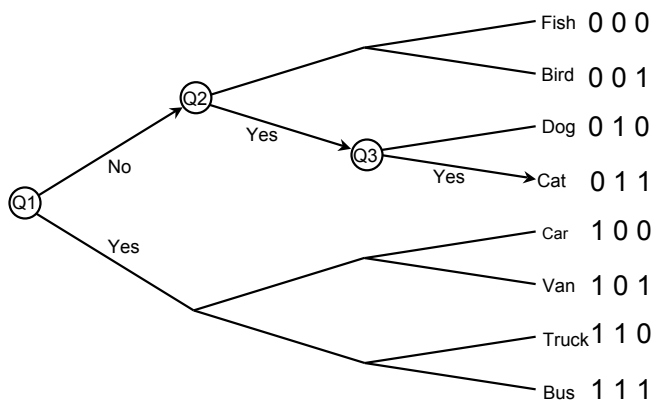


Figure 1.3.: The game of 20 questions, abbreviated to 3 questions here. Given an opponent who has one of 8 words in mind, each yes/no question halves the number of remaining possible words. Each binary number on the right summarises the answers required to arrive at one word (no=0 and a yes=1).

1.4. A Million Answers to Twenty Questions

In this game, your opponent has a vocabulary of exactly 8 words, and you know which words they are. Your first question (Q1) could be, “Is it inanimate?”, and the answer should halve the number of possible words to 4, and lead you to your second question (Q2). If your second question (Q2) is, “Is it a mammal?”, then the answer should again halve the number of possible words, which leads to your third question (Q3). By the time you arrive at Q3, there are just two possible words left, and after you have asked the third question (e.g. “Is it cat?”), your opponent’s yes/no response should lead you to the correct answer. In summary, you have asked three questions, and in so doing, you have excluded all but one out of eight possible words.

More realistically, let’s assume your opponent has the same vocabulary as you do (most of us have similar vocabularies, so this assumption is not unreasonable). For simplicity, let’s assume this vocabulary contains exactly 1,048,576 words. Armed with this knowledge, in principle, each of your questions can be chosen to halve the number of remaining possible words. So, in an ideal world, your first question should halve the number of possible words to 524,288. Your next question should halve this to 262,144 words, and so on. By the time you get to the 19th question there should be just two words left, and after the 20th question, there should be only one word left.

The reason this works out so neatly is because 20 questions allows you to choose from amongst exactly $1,048,576 = 2^{20}$ equally probable words (i.e. about one million). Thus, the 20 bits of information you have acquired with your questions provide you with the ability to narrow down the range of possible words that your opponent has chosen from about 1 million to just one. In other words, 20 questions allows you to find the correct word out of about a million possible words.

Adding one more question would not only create a new game ‘21 questions’, it would also double the number of possible words (to about 2 million) that you could narrow down to one. By extension, each additional question allows you to acquire up to one more bit of information, and can therefore double the initial number of words. In principle, a game of ‘40 questions’ would allow you to acquire 40 bits

1 What Is Information?

of information, which would allow you to find one out of $2^{40} \approx 10^{12}$ (a million million) words.

In terms of the navigation example above, 40 bits would allow you to navigate 40 forks in the road, and would therefore permit you to choose one out of about a trillion possible routes (which we will round down to one trillion for simplicity). Because each route leads to one out of a trillion different destinations, 40 bits would allow you to find one out of 1 trillion destinations. So the next time you arrive at your destination after a journey that involved 40 decisions, remember that, you have avoided arriving at a trillion-minus-one incorrect destinations.

1.5. Information, Bits and Binary Digits

Despite the fact that the word *bit* is derived from *binary digit*, there is a subtle, but vital, difference between them. A binary digit is a zero or a one, but it is not information *per se*. In contrast, a bit is a definite *amount of information*, which can be conveyed by any pair of equally probable symbols, such as a binary digit (0/1), or words (e.g. yes/no). Bits and binary digits are different kinds of entities, so to confuse one with the other is a category error.

Even though a bit is a unit of information, it is not anything like a quantum of information because it is not the smallest amount of information that can exist. In fact, as we shall see in Chapter 2, a binary digit conveys an amount of information which varies continuously between a minimum of zero and a maximum of one bit of information.

To give an extreme example, if you already know that you should take the left-hand road from point A in Figure 1.2, and if I show you the binary digit 0 (=go left) then I have given you a binary digit, but you have gained no information. In contrast, suppose you already know that there is a 71% probability that you should take the left-hand road. This means that you must already have some information. If you had one bit of information then you would know which road to take; so you must already have less than one bit. Now, if I show you the binary digit 0 (left) then I have given you a binary digit, but, because you already have some information, you have gained less than one bit of information. As we shall see in Section 5.8, even though you have been given one binary digit, you have gained only *half a bit* of information.

So, even though I cannot give you half a binary digit, when I give you a binary digit, this can provide you with half a bit of information.

The distinction between binary digits and bits is often ignored, with Pierce's book³⁵ being a notable exception. Even some of the best textbooks use the terms 'bit' and 'binary digit' interchangeably. This causes few problems for experienced readers, who can interpret the term 'bit' as meaning a binary digit or a bit's worth of information, according to its context. But for novices, the failure to respect this distinction is a source of genuine confusion.

Sadly, in modern usage, the terms bit and binary digit have become synonymous, and MacKay(2003)²⁹ proposed that the unit of information should be called the *Shannon*.

Key point. A binary digit can adopt one of two possible values (0 or 1), whereas a bit is an *amount of information* which can adopt any value between zero and one.

1.6. Example 1: Telegraphy

Suppose you have just discovered that, if you hold a compass next to a wire then the compass needle changes position when you pass a current through the wire. If the wire is long enough to connect two towns like London and Manchester then a current initiated in London can deflect a compass needle held near to the wire in Manchester.

Naturally, you would like to use this new technology to send messages in the form of individual letters. Sadly, the year is 1820, so you would have to wait over 100 years for Shannon's book to be published. Undeterred, you forge ahead. Let's say you want to send only upper case letters, to keep matters simple. So you set up 26 electric lines, one per letter from *A* to *Z*, with the first line being *A*, the second line being *B*, and so on. Each line is set up next to a compass, which is kept some distance from all the other lines to prevent each line from deflecting more than one compass.

In London, each line is labelled with a letter, and the corresponding line is labelled with the same letter in Manchester. For example, if you want to send the letter *D*, you press a switch on the fourth line in London, which sends an electric current to Manchester along the wire

1 What Is Information?

which is next to the compass labelled with the letter *D*. Of course, lines fail from time to time, and it is about 200 miles from London to Manchester, so finding the location of the break in a line is difficult and expensive. Naturally, if there were fewer lines then there would be fewer failures.

With this in mind, Cooke and Wheatstone devised a complicated two-needle system, which could send only 23 different letters. Despite the complexity of their system, it famously led to the arrest of a murderer. On the first of January 1845, John Tawell poisoned his mistress, Sarah Hart, in a place called Salt Hill in the county of Berkshire, before escaping on a train to Paddington station in London. In order to ensure Tawell was arrested when he reached his destination, the following telegraph was sent to London:

A MURDER HAS GUST BEEN COMMITTED AT SALT
HILL AND THE SUSPECTED MURDERER WAS SEEN
TO TAKE A FIRST CLASS TICKET TO LONDON BY
THE TRAIN WHICH LEFT SLOUGH AT 742 PM HE IS
IN THE GARB OF A KWAKER ...

The unusual spellings of the words JUST and QUAKER were a result of the telegrapher doing his best in the absence of the letters J, Q and Z in the array of 23 letters before him. As a result of this telegram, Tawell was arrested, and subsequently hanged for murder. The role of Cooke and Wheatstone’s telegraph in Tawell’s arrest was widely reported in the press, and established the practicality of telegraphy.

A	• -	J	• - - -	S	• • •
B	- • • •	K	- • -	T	-
C	- • - •	L	• - • •	U	• • -
D	- • •	M	- -	V	• • -
E	•	N	- •	W	• - -
F	• • - •	O	- - -	X	- • • -
G	- - •	P	• - - •	Y	- • - -
H	• • • •	Q	- - • -	Z	- - • •
I	• •	R	• - •		

Table 1.2.: Morse code. Common letters (e.g. E) have the shortest codewords, whereas uncommon letters (e.g. Z) have the longest codewords.

In the 1830s, Samuel Morse and Alfred Vail developed the first version of (what came to be known as) the *Morse code*. Because this specified each letter as dots and dashes, it could be used to send messages over a single line.

An important property of Morse code is that it uses short codewords for the most common letters, and long codewords for less common letters, as shown in Table 1.2. In order to find out which letters were most common, Morse adopted a simple strategy. Reasoning that newspaper printers would have only as many copies of each letter as required, he went to a printer's workshop, and counted how many copies of each letter were there. As a result, the most common letter E is specified as a single dot, whereas the rare J is specified as a dot followed by three dashes. This is important because it permits an efficient use of the communication channel (a single wire). This is a theme to which we will return many times, and it raises the question: how could we tell if a communication channel is being used as efficiently as possible?

1.7. Example 2: Binary Images

The internal structure of most images is highly predictable. For example, most of the individual *picture elements* or *pixels* in the image of stars in Figure 1.4 are black, with an occasional white pixel, a star. Because almost all pixels are black, it follows that most pairs of adjacent pixels are also black, which is what makes the image's internal structure predictable. If this picture were taken by the orbiting Hubble telescope then its predictable structure would ensure it could be efficiently transmitted to Earth.

Suppose you were in charge of writing the computer code which conveys the information in Figure 1.4 from the Hubble telescope to Earth. For example, you could naively send the value of each pixel; let's call this method A. Because there are only two values in this particular image (black and white), you could choose to indicate the colour black with the binary digit 0, and the colour white with a 1. You would therefore need to send as many 0s and 1s as there are pixels in the image. For example, if the image was 100×100 pixels the you would need to send ten thousand 0s or 1s in order for the image to be reconstructed on Earth. Because almost all the pixels are black, you

1 What Is Information?



Figure 1.4.: The night sky. Each pixel contains one of just two values.

would send sequences of hundreds of 0s interrupted by the occasional 1. It is not hard to see that this is a wasteful use of the expensive satellite communication channel. How could it be made more efficient?

Another method consists of sending only the locations of the white pixels (method B). This would yield a code that looked a bit like this $[(19, 13), (22, 30), \dots]$, where each pair of numbers represents the row and column of a white pixel.

Yet another method consists of concatenating all of the rows of the image, and then sending the number of black pixels that occur before the next white pixel (method C). So, if the number of black pixels that precede the first white pixel is 13, and there are 9 pixels before the next white pixel then the numbers in the first row of the image begin with 000000000000010000000001..., and the code for communicating this would be $[13, 9, \dots]$, which is clearly more compact than the 24 zeros and ones in the first part of the first row of the image.

Notice that method A consists of sending the image itself, whereas methods B and C do not send the image, but they do send all of the *information* required to reconstruct the image on Earth. Crucially, the end results of all three methods are identical, and it is only the efficiency of the methods that differs.

In fact, the most efficient method depends on the structure of the image. This can be seen if we take an extreme example which consists of just one white pixel in the centre of the image. For this image, method A is clearly fairly useless, because it would require 10,000 binary values to be sent. Method B would consist of two numbers, $(50, 50)$, and



Figure 1.5.: In a binary image each pixel has 1 out of 2 possible grey-levels.

method C would consist of a single number 5,050. If we ignore the brackets and commas then we end up with 4 decimal digits for both methods B and C. So these methods seem to be equivalent, at least for the example considered here.

For other images, with other structures, different *encoding methods* will be more or less efficient. For example, Figure 1.5 contains just two grey-levels, but these occur in large regions of pure black or pure white. In this case, it seems silly to use method B to send the location of every white pixel, because so many of them occur in long runs of white pixels. This observation makes method C seem to be an obvious choice; but with a slight change. Because there are roughly equal numbers of black and white pixels, which occur in regions of pure black or pure white, we could just send the number of pixels until the next change from black to white or from white to black. This is known as *run-length encoding*. For example, if the distance from the first black pixel in the middle row to the first white pixel is 87 pixels (which is the girl's hair), and the distance from here to the next black pixel is 31 pixels, and the distance to the next white pixel is 18 pixels, then this part of the encoded image would look like this $[\dots, 87, 31, 18, \dots]$.

1.8. Example 3: Grey-Level Images

Suppose we wanted to transmit an image of 100×100 pixels, in which each pixel could adopt more than two possible grey-level values. A reasonable number of grey-levels turns out to be 256, as shown in Figure 1.6a. As before, there are large regions that look as if they

1 What Is Information?

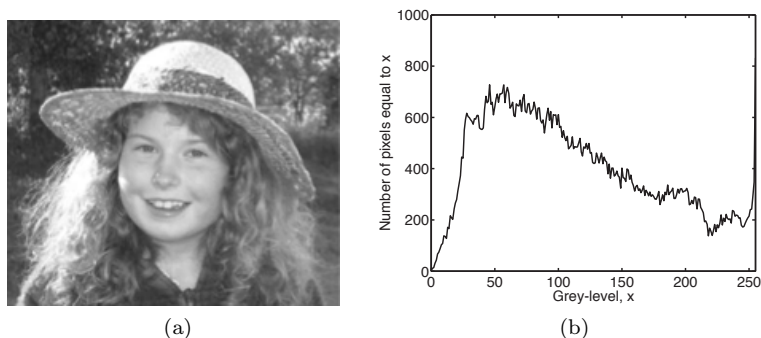


Figure 1.6.: Grey-level image. a) An image in which each pixel has one out of 256 possible grey-levels, between 0 and 255, each of which can be represented by a binary number with 8 binary digits (e.g. $255=11111111$). b) Histogram of grey-levels in the picture.

contain the same grey-level. In fact, each such region contains grey-levels which are similar, but not identical, as shown in Figure 1.7. The similarity between nearby pixel values means that adjacent pixel values are not *independent* of each other, and that the image has a degree of *redundancy*. How can this observation be used to encode the image?

One method consists of encoding the image in terms of the differences between adjacent pixel grey-levels. For brevity, we will call this *difference coding* (more complex methods exist, but most are similar in spirit to this simple method). In principle, pixel differences could be measured in any direction within the image, but for simplicity, we concatenate consecutive rows to form a single row of 10,000 pixels, and then take the difference between adjacent grey-levels. We can see the result of difference coding by ‘un-concatenating’ the rows to reconstitute an image, as shown in Figure 1.8a, which looks like a badly printed version of Figure 1.6a. As we shall see, both images contain the same amount of information.

If adjacent pixel grey-levels in a given row are similar then the difference between grey-levels tends to be close to zero. In fact, a histogram of difference values shown in Figure 1.8b shows that the most common difference values are indeed close to zero, and only rarely greater than ± 63 . Thus, using difference coding, we could represent almost all of the 9,999 difference values in Figure 1.8a as a number between ± 63 .

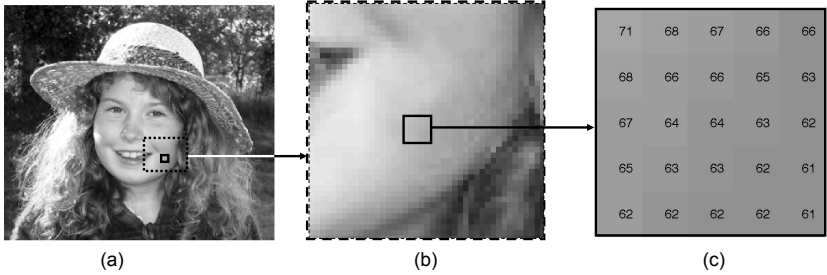


Figure 1.7.: Adjacent pixels tend to have similar grey-levels, so the image has a large amount of redundancy, which can be used for efficient encoding. a) Grey-level image. b) Magnified square from a. c) Magnified square from b, with individual pixel grey-levels indicated.

In those rare cases where there is a grey-level difference larger than ± 63 , we could list these separately as each pixel's (row, column) location (as 2×7 binary digits), and its grey-level (8 binary digits). Most coding procedures have special housekeeping fragments of computer code to deal with things like this, but these account for a negligible percentage of the total storage space required. For simplicity, we will assume that this percentage is zero.

At first, it is not obvious how difference coding represents any saving over simply sending the value of each pixel's grey-level. However, because these differences are between -63 and $+63$, they span a range of 127 different values, i.e. $[-63, 62, \dots, 0, \dots, 62, 63]$. Crucially, any number in this range can be represented using 7 binary digits because $7 = \log 128$ (leaving one spare value).

In contrast, if we were to send each pixel's grey-level in Figure 1.6a individually then we would need to send 10,000 grey-levels. Because each grey-level could be any value between 0 and 255, we would have to send 8 binary digits ($8 = \log 256$) for each pixel.

Once we have encoded an image into 9,999 pixel difference grey-levels $(d_1, d_2, \dots, d_{9999})$, how would we use these to reconstruct the original image? If the difference d_1 between the first pixel grey-level x_1 and the second pixel grey-level x_2 is, say, $d_1 = (x_2 - x_1) = 10$ grey-levels, and if the grey-level of x_1 is 5 then the original grey-level of x_2 can be obtained by adding 10 to x_1 ; that is, $x_2 = x_1 + d_1$ so $x_2 = 5 + 10 = 15$. This process is then continued for the third pixel ($x_3 = x_2 + d_2$), and

1 What Is Information?

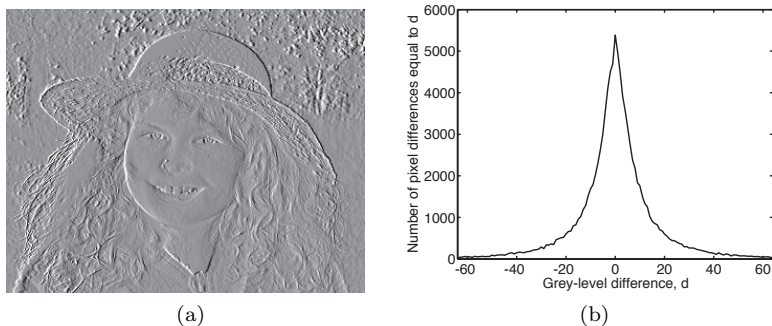


Figure 1.8.: Difference coding. a) Each pixel grey-level is the difference between adjacent horizontal grey-levels values in Figure 1.6a (grey = zero difference). b) Histogram of grey-level differences between adjacent pixel grey-levels in Figure 1.6a. Only differences between ± 63 are plotted.

so on. Thus, provided we know the grey-level of the first pixel in the original image (which can be encoded as 8 binary digits), we can use the pixel difference grey-levels to recover the grey-level of every pixel in the original image. The fact that we can reconstruct the original image (Figure 1.6a) from the grey-level differences (Figure 1.8a) proves that they both contain exactly the same amount of *information*.

Let's work out the total saving using this difference coding method. The naive method of sending all pixel grey-levels, which vary between 0 and 255, would need 8 binary digits per pixel; which requires a total of 80,000 binary digits. Using difference coding, we would need 7 binary digits per difference value; making a total of 70,000 binary digits. Therefore, using difference coding provides a saving of 10,000 binary digits, or 12.5%.

In practice, a form of difference coding is used to reduce the amount of data required to transmit voices over the telephone, where it is known as *differential pulse code modulation*. Using the differences between consecutive values, a voice signal which would otherwise require 8 binary digits per value can be transmitted with just 5 binary digits.

As we shall see in subsequent chapters, a histogram of data values (e.g. image grey-levels) can be used to find an upper bound for the average amount of information each data value could convey. Accordingly, the histogram (Figure 1.6b) of the grey-levels in Figure 1.6a defines an upper bound of 7.84 bits/pixel. In contrast, the

histogram (Figure 1.8b) of the grey-level differences in Figure 1.8a defines an upper bound of just 5.92 bits/pixel.

Given that the images in Figures 1.6a and 1.8a contain the same amount of information, and that Figure 1.8a contains no more than 5.92 bits/pixel, it follows that Figure 1.6a cannot contain more than 5.92 bits/pixel. This matters is because Shannon's work guarantees that, if each pixel's grey-level contains an average of 5.92 bits then we should be able to represent Figure 1.6a using no more than 5.92 binary digits per pixel. But this still represents an upper bound. As we shall see, the smallest number of binary digits required to represent each pixel is equal to the number of bits implicit in each pixel. So, what we really want to know is, how much information does each pixel contain?

This is a hard question, but we can get a better idea of the answer by comparing the amount of computer memory required to represent the image on a computer screen with the amount of memory required to store that image on the hard-drive. Image files are usually stored in compressed form, where the compression method is indicated by the file name extension (e.g. '.tiff'). If the compression method used does not throw away any information then it is *lossless*, otherwise it is *lossy*.

Now, the image in Figure 1.6a is actually 344 by 299 pixels, where each pixel grey-level is between 0 and 255, which can be represented as 8 binary digits (because $2^8 = 256$), or 1 *byte*. This amounts to a total of 102,856 pixels, each of which is represented on a computer screen as 1 byte. However, when the file containing this image is inspected, it is found to contain a mere 45,180 bytes. Apparently, the image can be compressed by a factor of 2.28(= 102856/45180) without any loss of information. This means that the information implicit in each pixel, which requires 8 binary digits for it to be displayed on a screen, can be represented with about 4 binary digits on a computer's hard-drive.

Thus, even though each pixel can adopt one out of 256 possible grey-levels, and is displayed using 8 binary digits of computer memory, the grey-level of each pixel can be stored using about 4 binary digits. This is important, because it implies that each set of 8 binary digits used to display each pixel in Figure 1.6a contains an average of only 4 bits, and therefore each binary digit contains only *half a bit* of information.

1.9. Summary

From navigating a series of forks in the road, and playing the game of ‘Twenty Questions’, we have seen how making binary choices requires information in the form of simple yes/no answers. These choices can also be used to choose from amongst a set of letters, and can therefore be used to send typed messages along telegraph wires.

We found that increasing the number of choices from 2 (forks in the road) to 26 (letters) to 256 (pixel grey-levels) allowed us to transmit whole images down a single wire as a sequence of binary digits. In each case, the redundancy of the data in a message allowed it to be compressed before being transmitted. This redundancy emphasizes a key point: a binary digit is *not* the same as a bit of information.

Whereas a binary digit is a zero or a one, a bit is the most information that can be conveyed by a binary digit (or any other binary variable, like yes/no). The information conveyed by a binary digit is only equal to one bit if the two alternatives under consideration are equally probable.

So, what is information? It is what remains after every iota of natural redundancy has been squeezed out of a message, and after every aimless syllable of noise has been removed. It is the unfettered essence that passes from computer to computer, from satellite to Earth, from eye to brain, and (over many generations of natural selection) from the natural world to the collective gene pool of every species.

Bibliography

- [1] Applebaum, D. (2008). *Probability and Information An Integrated Approach, 2nd Edition*. Cambridge University Press, Cambridge.
- [2] Avery, J. (2003). *Information Theory and Evolution*. World Scientific Publishing.
- [3] Baeyer, H. (2005). *Information: The New Language of Science*. Harvard University Press.
- [4] Baldwin, J. (1896). A new factor in evolution. *The American Naturalist*, 30(354):441–451.
- [5] Barlow, H. (1961). Possible principles underlying the transformation of sensory messages. *Sensory Communication*, pages 217–234.
- [6] Bennett, C. (1987). Demons, engines and the second law. *Scientific American*, 257(5):108–116.
- [7] Bérut, A., Arakelyan, A., Petrosyan, A., Ciliberto, S., Dillenschneider, R., and Lutz, E. (2012). Experimental verification of Landauer’s principle linking information and thermodynamics. *Nature*, 483(7388):187–189.
- [8] Bialek, W. (2012). *Biophysics: Searching for Principles*. Princeton University Press.
- [9] Bishop, C. (2006). *Pattern Recognition and Machine Learning*. Springer.
- [10] Cover, T. and Thomas, J. (1991). *Elements of Information Theory*. New York, John Wiley and Sons.
- [11] Darwin, C. (1859). *On the origin of species by means of natural selection, or the preservation of favoured races in the struggle for life*. 1st Edition, London, John Murray.
- [12] DeGroot, M. (1986). *Probability and Statistics, 2nd edition*. UK, Addison-Wesley.

Bibliography

- [13] Deutsch, D. and Marletto, C. (2014). Reconstructing physics: The universe is information. *New Scientist*, 2970:30.
- [14] Feynman, R., Leighton, R., and Sands, M. (1964). *The Feynman Lectures on Physics*. Basic Books.
- [15] Friston, K. (2010). The free-energy principle: a unified brain theory? *Nature Review Neuroscience*, 11(2):127–138.
- [16] Gabor, D. (1951). MIT Lectures.
- [17] Gatenby, R. and Frieden, B. (2013). The critical roles of information and nonequilibrium thermodynamics in evolution of living systems. *Bulletin of Mathematical Biology*, 75(4):589–601.
- [18] Gleick, J. (2012). *The Information*. Vintage.
- [19] Guizzo, E. (2003). The essential message: Claude Shannon and the making of information theory. <http://dspace.mit.edu/bitstream/handle/1721.1/39429/54526133.pdf>. *Massachusetts Institute of Technology*.
- [20] Holland, J. (1992). *Adaptation in Natural and Artificial Systems*. MIT Press.
- [21] Jaynes, E. and Bretthorst, G. (2003). *Probability Theory: The Logic of Science*. Cambridge University Press, Cambridge.
- [22] Jessop, A. (1995). *Informed Assessments: An Introduction to Information, Entropy and Statistics*. Ellis Horwood, London.
- [23] Kolmogorov, A. (1933). *Foundations of the Theory of Probability*. Chelsea Publishing Company, (English translation, 1956).
- [24] Kostal, L., Lansky, P., and Rospars, J.-P. (2008). Efficient olfactory coding in the pheromone receptor neuron of a moth. *PLoS Computational Biology*, 4(4).
- [25] Landauer, R. (1961). Irreversibility and heat generation in the computing process. *IBM Journal of Research and Development*, 5:183–191.
- [26] Laughlin, S. (1981). A simple coding procedure enhances a neuron’s information capacity. *Z Naturforsch*, 36c:910–912.
- [27] Lemon, D. (2013). *A Student’s Guide to Entropy*. Cambridge University Press, Cambridge.
- [28] Mackay, A. (1967). Optimization of the genetic code. *Nature*, 216:159–160.

- [29] MacKay, D. (2003). *Information theory, inference, and learning algorithms*. Cambridge University Press.
- [30] Nemenman, I., Lewen, G., Bialek, W., and de Ruyter van Steveninck, R. (2008). Neural coding of natural stimuli: Information at sub-millisecond resolution. *PLoS Computational Biology*, 4(3).
- [31] Nemenman, I., Shafee, F., and Bialek, W. (2002). Entropy and inference, revisited. In *TG Dietterich, S Becker, and Z Ghahramani, editors, Advances in Neural Information Processing Systems 14, Cambridge, MA, 2002*.
- [32] Nyquist, H. (2002). Certain topics in telegraph transmission theory. *Proceedings of the IEEE*, 90(2):280–305.
- [33] Olshausen, B. and Field, D. (1996). Natural image statistics and efficient coding. *Network: Computation in Neural Systems*, 7:333–339.
- [34] Paninski, L. (2003). Estimation of entropy and mutual information. *Neural Computation*, 15(6):1191–1253.
- [35] Pierce, J. (1961 reprinted by Dover 1980). *An introduction to information theory: symbols, signals and noise*. Dover (2nd Edition).
- [36] Reza, F. (1961). *Information Theory*. New York, McGraw-Hill.
- [37] Rieke, F., Bodnar, D., and Bialek, W. (1995). Naturalistic stimuli increase the rate and efficiency of information transmission by primary auditory afferents. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 262(1365):259–265.
- [38] Rieke, F., Warland, D., van Steveninck, R., and Bialek, W. (1997). *Spikes: Exploring the Neural Code*. MIT Press, Cambridge, MA.
- [39] Schneider, T. D. (2010). 70% efficiency of bistate molecular machines explained by information theory, high dimensional geometry and evolutionary convergence. *Nucleic acids research*, 38(18):5995–6006.
- [40] Schürmann, T. and Grassberger, P. (1996). Entropy estimation of symbol sequences. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 6(3):414–427.
- [41] Sengupta, B., Stemmler, M., and Friston, K. (2013). Information and efficiency in the nervous system - a synthesis. *PLoS Computational Biology*.
- [42] Shannon, C. (1948). A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423.

Bibliography

- [43] Shannon, C. (1951). Prediction and entropy of printed English. *Bell System Technical Journal*, 30:47–51.
- [44] Shannon, C. and Weaver, W. (1949). *The Mathematical Theory of Communication*. University of Illinois Press, Urbana and Chicago.
- [45] Stone, J. (2012). *Vision and Brain: How we perceive the world*. MIT Press.
- [46] Stone, J. (2013). *Bayes' Rule: A Tutorial Introduction to Bayesian Analysis*. Sebtel Press, Sheffield, England.
- [47] Wallis, K. (2006). A note on the calculation of entropy from histograms. Technical report, University of Warwick, UK.
- [48] Watkins, C. (2008). Selective breeding analysed as a communication channel: Channel capacity as a fundamental limit on adaptive complexity. *Symbolic and Numeric Algorithms for Scientific Computing: Proceedings of SYNASC'08*, pages 514–518.
- [49] Zhaoping, L. (2014). *Understanding Vision: Theory, Models, and Data*. OUP Oxford.

Index

- alphabet, 23, 203
- average, 203
- Baldwin effect, 191
- ban, 41
- bandwidth, 154
- Barlow, H, 192
- Bayes' rule, 113, 114, 145, 203, 221, 223, 224
- Bennett, CH, 179
- biased entropy, 113
- binary
 - digit, 4, 203
 - digits vs bits, 10
 - number, 6, 49, 204
 - string, 204
 - symmetric channel, 94, 204
- bit, 3, 4, 41, 124, 204
- block codes, 97
- byte, 19, 204
- capacity, 29, 44, 100, 147, 204
- central limit theorem, 123, 226
- chain rule for entropy, 143, 228
- channel, 107, 130, 204
 - fixed range, 159
 - Gaussian, 148
 - noiseless, 45
 - noisy, 25, 76
- channel capacity, 29
- code, 204
 - error correcting, 97
- codebook, 29, 105, 157, 204
- codeword, 27, 49, 204
- coding
 - block, 97
 - efficiency, 51, 54, 204
 - Huffman, 55
 - prefix, 57
 - Shannon-Fano , 59
- communication channel, 26
- compression, 28, 47
 - lossless, 28
 - lossy, 28
- computational neuroscience, 3, 192
- conditional
 - entropy, 205
 - probability, 204, 223
 - probability distribution, 135
- correlation, 158
- correlation length, 63
- counting argument, 68
- cross-talk, 91
- cumulative density function, 155, 165, 198, 226
- Deutsch, D, 147
- die
 - 11-sided, 54
 - 16-sided, 40
 - 6-sided, 50
 - 6-sided pair, 52
 - 8-sided, 37, 48
 - 9.65-sided, 55
- difference coding, 17
- differential entropy, 112, 205
- discrete, 205

Index

- discrete variable, 22
- disorder, 170
- efficient code, 50, 204
- efficient coding hypothesis, 3, 192, 199
- Einstein, A, 1
- encoding, 15, 28, 52, 117, 130, 160, 161, 196, 205
- ensemble, 61, 174, 205
- entropy, 21, 33, 36, 205
 - biased estimate, 113
 - differential, 111
 - exponential distribution, 120
 - Gaussian distribution, 121
 - information, 169
 - maximum, 117
 - negative, 120
 - prior, 113
 - rate, 206
 - Shannon, 169
 - thermodynamic, 169
 - uniform distribution, 118
- equivocation, 206
- ergodic, 206
- error correcting code, 97
- evolution, 191
- expected value, 37, 206
- exponential distribution, 120
- fan diagram, 91
- Feynman, R, 169, 183
- fly, 117, 166, 196
- Gaussian distribution, 121, 225
- genetic algorithm, 187
- genome, 186
- Greene, B, 179
- histogram, 108, 217, 218
- Holland's schema theorem, 188
- Holland, J, 187
- Huffman code, 55
- Huffman, D, 55
- iid, 41, 206
- independence, 16, 30, 33, 41, 44, 55, 59, 82, 83, 131, 158, 206
- Infeld, L, 1
- information, 10, 30, 40, 124, 176
 - Shannon, 30
- information entropy, 169
- instantaneous code, 208
- integration, 206
- Kolmogorov complexity, 73, 207
- Kolmogorov, A, 73
- Kullback-Leibler divergence, 145, 207
- Landauer limit, 176
- Laughlin, S, 196
- law of large numbers, 69, 207
- likelihood function, 211
- logarithm, 7, 30, 41, 207, 215
- lossless compression, 28, 55
- lossy compression, 28
- macrostate, 171
- marginal
 - definition, 207
- marginal probability distribution, 131
- Marletto, C, 147
- maximum entropy distribution, 117
- mean, 207
- message, 25, 207
- microstate, 171
- monotonic, 115, 144, 160, 207
- Morse code, 12, 13, 68
- Morse, S, 12
- MPEG, 184
- mutual information, 75, 81, 85, 86, 90, 144
 - channel capacity, 146
 - continuous, 134, 139

- Gaussian channel, 148
- relative entropy, 145
- nats, 42
- natural logarithms, 173, 216
- natural selection, 187
- noise, 2, 75, 85, 89, 91, 105, 207
- non-computable, 73
- outcome, 25, 207
- outcome value, 25, 207
- outer product, 83, 131, 208
- parity, 98, 208
- posterior distribution, 211
- Pratchett, T, 182
- precision, 208
- prefix code, 57, 208
- prior distribution, 211
- prior for entropy, 113
- probability
 - conditional, 223
 - definition, 208
 - density, 109, 218
 - density function, 108, 208, 217
 - distribution, 208
 - function, 76, 209
 - mass function, 209
 - rules of, 221
- product rule, 221, 223
- random variable, 22, 209, 211
- redundancy, 16, 28, 97, 100, 184, 185, 209
- relative entropy, 145, 209
- relative frequency, 21, 60, 70, 113, 209
- residual uncertainty, 125
- run-length encoding, 15
- second law of thermodynamics, 178
- self-punctuating code, 208
- sex, 191
- Shakespeare, W, 43
- Shannon
 - entropy, 169
 - information unit, 11, 42
- Shannon, C, 1, 4, 21, 41, 129
- Shannon-Fano coding, 59
- signal to noise ratio, 150
- source, 25
- source coding theorem, 47
- spike train, 194
- standard deviation, 121, 210
- stationary source, 41, 68, 210
- sum rule, 221, 223
- surprisal, 30
- surprise, 32
- symbol, 25, 26
 - definition, 210
- telegraphy, 11
- terminology, 25
- theorem, 210
 - noisy channel coding, 100
 - noisy channel coding theorem, 101, 105
 - schema, 188
 - source coding, 47
- thermodynamic entropy, 169
- transmission efficiency, 97
- uncertainty, 33, 40, 61, 75, 88, 93, 124, 210
- Vail, A, 12
- variable, 210
- variance, 122, 210
- vector, 152
- Weaver, W, 75
- Wheeler, J, 107

About the Author.

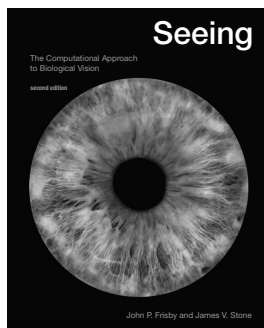
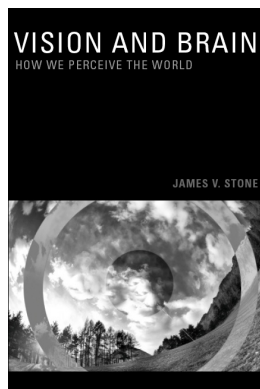
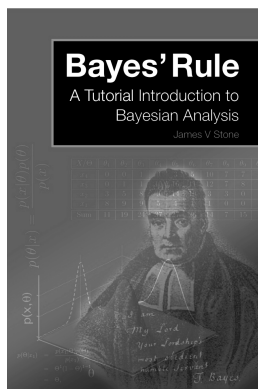
Dr James Stone is a Reader in Computational Neuroscience at the University of Sheffield, England. Previous books are listed below.

Bayes' Rule: A Tutorial Introduction to Bayesian Analysis,
JV Stone, Sebtel Press, 2013.

Vision and Brain: How We Perceive the World, JV Stone,
MIT Press, 2012.

Seeing: The Computational Approach to Biological Vision,
JP Frisby and JV Stone, MIT Press, 2010.

Independent Component Analysis: A Tutorial Introduction,
JV Stone, MIT Press, 2004.



Information Theory: A Tutorial Introduction

Originally developed by Claude Shannon in the 1940s, information theory laid the foundations for the digital revolution, and is now an essential tool in telecommunications, genetics, linguistics, brain sciences and deep space communication. In this richly illustrated book, accessible examples are used to introduce information theory in terms of everyday games like ‘20 questions’, before more advanced topics are explored. The online MatLab and Python computer programs give hands-on experience of information theory in action, and PowerPoint slides provide support for teaching. Written in a tutorial style, with a comprehensive glossary and tutorial appendices on basic maths, this text represents an ideal primer for novices who wish to become familiar with the essential principles and applications of information theory.

Features

- Tutorial style of writing ideal for novices.
- Intuitive diagrams used to express key concepts.
- Text boxes summarise main points.
- Comprehensive glossary of technical terms.
- Tutorial appendices explain basic mathematical concepts.
- Example code in Python and MatLab.
- PowerPoint slides of figures available online.

